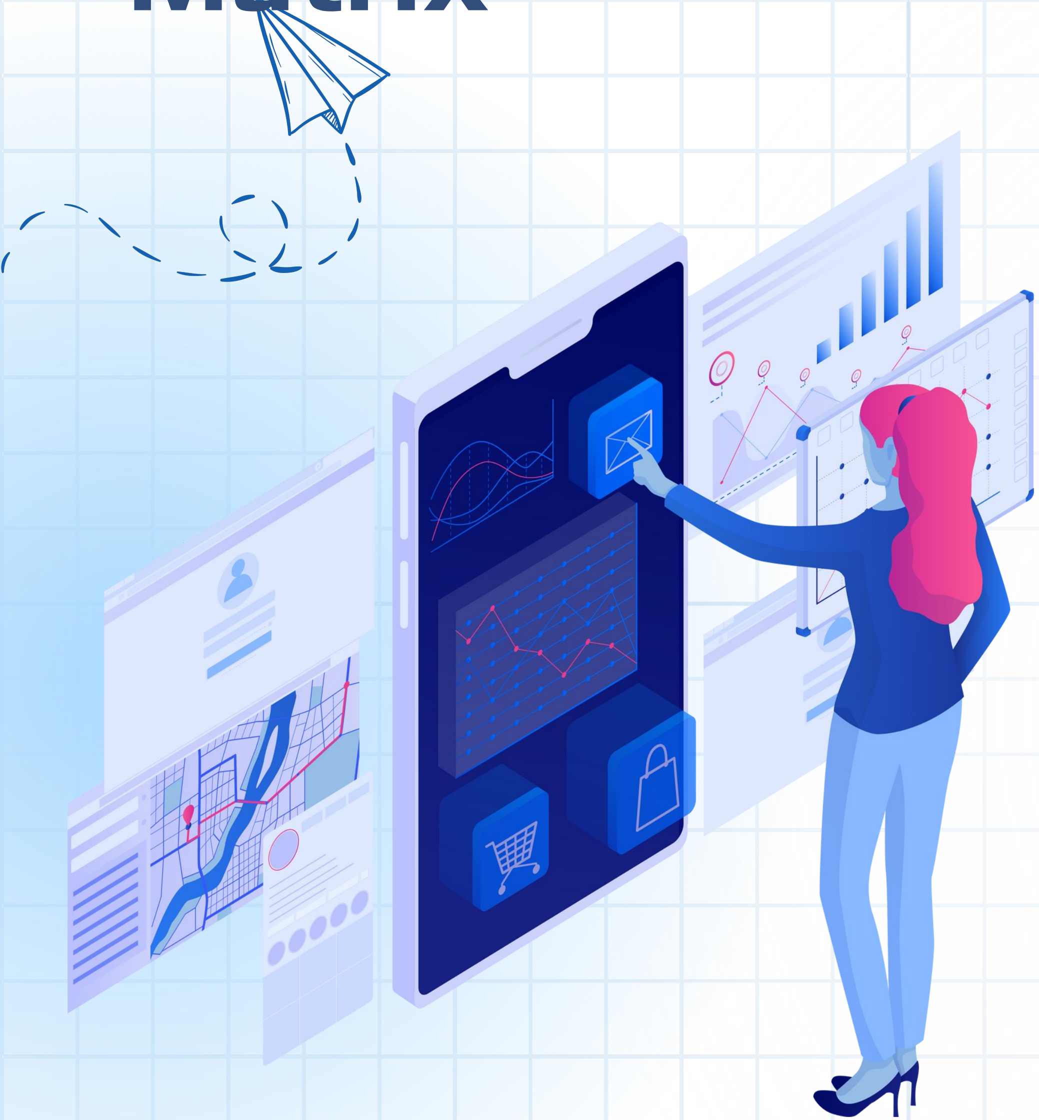
BS./BSc.

in Applied AI and Data Science

Linear algebra and numerical analysis

Powered by Futurense

Module 2.3 Inverse and Rank of a Matrix



1. Symmetric and Square matrix
2. The determinants
3. Inverse of a Matrix
4. Rank of a matrix

Inverse matrix operations in daily life

Invertible matrices have many special properties, making them great to work with in numerous real-world applications.

Please see Examples below:

- ❖ For example, you might use linear equations to model supply and demand in **economics**. Using **inverse matrices**, you can efficiently solve for the unknown variables.
- ❖ Inverse matrices are crucial in **3D computer graphics** for transformations like rotations, translations, and scaling.
- ❖ Matrices are used in **encryption and decryption**. Inverse matrices play a vital role in decoding **encrypted messages**.
- ❖ **GPS systems** use matrices to calculate positions and distances. **Inverse matrices** are used in these calculations to solve for unknown coordinates.

Matrix Inverse worked out example

Example: Given A and C below, show that C is the inverse of A.

$$A = \begin{bmatrix} 1 & -3 & 0 \\ -1 & 2 & -2 \\ -2 & 6 & 1 \end{bmatrix}, \quad C = \begin{bmatrix} -14 & -3 & -6 \\ -5 & -1 & -2 \\ 2 & 0 & 1 \end{bmatrix}$$

Matrix Inverse worked out example

Solution:

Solution. Compute $\mathbf{AC}$:

$$\mathbf{AC} = \begin{bmatrix} 1 & -3 & 0 \\ -1 & 2 & -2 \\ -2 & 6 & 1 \end{bmatrix} \begin{bmatrix} -14 & -3 & -6 \\ -5 & -1 & -2 \\ 2 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Compute $\mathbf{CA}$:

$$\mathbf{CA} = \begin{bmatrix} -14 & -3 & -6 \\ -5 & -1 & -2 \\ 2 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & -3 & 0 \\ -1 & 2 & -2 \\ -2 & 6 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Therefore, by definition $\mathbf{C} = \mathbf{A}^{-1}$.

Example of solving a linear system of Equations

Example: solve the linear system Ax = b if

$$\mathbf{A} = \begin{bmatrix} 1 & -3 & 0 \\ -1 & 2 & -2 \\ -2 & 6 & 1 \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} 1 \\ -3 \\ -1 \end{bmatrix}.$$

$$\mathbf{A}^{-1} = \begin{bmatrix} -14 & -3 & -6 \\ -5 & -1 & -2 \\ 2 & 0 & 1 \end{bmatrix}.$$

Example of solving a linear system of Equations

Solutions: The unique solution to the linear system Ax = b is

$$A^{-1}Ax = A^{-1}b$$
$$\Rightarrow I_n x = A^{-1}b$$
$$\Rightarrow x = A^{-1}b.$$

Therefore, with $x = A^{-1}b$ we have that

$$A(A^{-1}b) = AA^{-1}b = I_n b = b$$

$$A^{-1}b = \begin{bmatrix} -14 & -3 & -6 \\ -5 & -1 & -2 \\ 2 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ -3 \\ -1 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}$$

Theorem based on inverse of Matrix operations

(1) The matrix $\mathbf{A}^{-1}$ is invertible and its inverse is $\mathbf{A}$:

$$(\mathbf{A}^{-1})^{-1} = \mathbf{A}.$$

(2) The matrix $\mathbf{AB}$ is invertible and its inverse is $\mathbf{B}^{-1}\mathbf{A}^{-1}$:

$$(\mathbf{AB})^{-1} = \mathbf{B}^{-1}\mathbf{A}^{-1}.$$

(3) The matrix $\mathbf{A}^T$ is invertible and its inverse is $(\mathbf{A}^{-1})^T$:

$$(\mathbf{A}^T)^{-1} = (\mathbf{A}^{-1})^T.$$

Coding Question

Given,

matrix1 = np.array([[1, 2], [3, 4]])
matrix2 = np.array([[1, 2, 3], [4, 5, 6], [7, 8,
9]])
matrix3 = np.array([[2, 0, 0], [0, 3, 0], [0, 0,
5]])
“”””

Computes the inverse of a square matrix.
Args: matrix: A NumPy array representing
the square matrix.
Returns: The inverse of the matrix, or None
if the matrix is singular (non-invertible). """"

Powered by Futurense

Solution

```
matrix2 = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]]) #singular m
inverse2 = invert_matrix(matrix2)

if inverse2 is not None:
  print("Inverse of matrix2:\n", inverse2)
else:
  print("matrix2 is not invertible.")

matrix3 = np.array([[2, 0, 0], [0, 3, 0], [0, 0, 5]])
inverse3 = invert_matrix(matrix3)
if inverse3 is not None:
    print("Inverse of matrix3:\n", inverse3)
else:
    print("matrix3 is not invertible.")
```

Solution explanation........

Explanation and Key Improvements:

1. NumPy's linalg.inv():

1. The code leverages NumPy's **np.linalg.inv()** function, which is the most efficient and reliable way to compute matrix inverses in Python.

2. Error Handling (Singular Matrices):

1. The try...except **np.linalg.LinAlgError block** is crucial. It gracefully handles cases where the input matrix is singular (non-invertible).
2. Singular matrices have a determinant of 0, and their inverses do not exist. Without error handling, **np.linalg.inv()** would raise an exception, causing the program to crash.
3. The function now returns None when a matrix is not invertible.

Relation between Rank and inverse of a Matrix

A matrix is invertible if and only if it has full rank. Here's a breakdown:

- **Invertible Matrix:** A square matrix A is invertible (or nonsingular) if matrix B exists such that AB = BA = I, where I is the identity matrix.
- **Rank of a Matrix:** The rank of a matrix is the maximum number of linearly independent rows or columns in the matrix.
- **Full Rank:** For an n x n square matrix, full rank means the rank equals n.

Maximum possible Rank of a Matrix

- Matrix rank **max(r)=rank(A)**

- A non-negative integer **(0,1,2,3,...)**

- Maximum possible rank

  max(r)=min(m,n)

  rank(A) <= min(m, n)

Where:

- **rank (A)** represents the rank of matrix A.
- **m** is the number of rows in matrix A.
- **n** is the number of columns in matrix A.
- **min(m, n)** returns the smaller value between m and n.

Linear Independence of a Vector Set

- Matrix rank **= rank(C(A)) or rank(R(A))**

Now that you know about matrix rank, you are ready to understand an algorithm for determining whether a set of vectors is **linearly independent**

The algorithm is straightforward: put the vectors into a matrix, compute the rank of the matrix, and then compare that rank to the maximum possible rank of that matrix(remember that this is min{M,N}

- *$r = M$: The vector set is linearly independent.*
- *$r < M$: The vector set is linearly dependent.*

Reasoning in previous algorithm of independence

- **The reasoning behind the previous algorithm should be clear: if the rank is smaller than the number of columns, then at least one column can be described as a linear combination of other columns, which is the definition of linear dependence.**

- **If the rank equals the number of columns, then each column contributes unique information to the matrix, which means that no column can be described as a linear combination of other columns.**

Question and solution

- **Determine the rank of each of the matrices shown below.**

$$\mathbf{A} = \begin{bmatrix} 1 \\ 2 \\ 4 \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} 1 & 3 \\ 2 & 6 \\ 4 & 12 \end{bmatrix}, \quad \mathbf{C} = \begin{bmatrix} 1 & 3.1 \\ 2 & 6 \\ 4 & 12 \end{bmatrix}, \quad \mathbf{D} = \begin{bmatrix} 1 & 3 & 2 \\ 6 & 6 & 1 \\ 4 & 2 & 0 \end{bmatrix}, \quad \mathbf{E} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}, \quad \mathbf{F} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

Solution

3 $r(\mathbf{A}) = 1, r(\mathbf{B}) = 1, r(\mathbf{C}) = 2, r(\mathbf{D}) = 3, r(\mathbf{E}) = 1, r(\mathbf{F}) = 0$

Recap

Understanding matrix rank is fundamental in linear algebra. Here are six key things to know:

1. Definition: Linear Independence

The rank of a matrix is the maximum number of linearly independent rows or columns within that matrix. This means that no row (or column) can be expressed as a linear combination of the others.

2. Maximum Rank: Dimensions Matter

For an $m \times n$ matrix (m rows, n columns), the maximum possible rank is the smaller of m and n. This is expressed as: $\text{rank}(A) \leq \min(m, n)$.

3. Zero Matrix: Rank Zero

The only matrix with a rank of 0 is the zero matrix, where all elements are zero.

4. Full Rank: Invertibility (for Square Matrices)

A square $n \times n$ matrix has "full rank" if its rank is n. A square matrix has full rank if and only if it is invertible (non-singular).

Coming up next...............

Module 3: **Methods of Finding Rank:**

There are several methods for determining a matrix's rank:

- **Echelon Form:** Reducing the matrix to echelon form and counting the number of non-zero rows.
- **Minors:** Finding the largest non-zero determinant of a square submatrix.
- **Using linear independence:** directly determining the number of independent rows or columns.
- **Eigenvalues and Eigenvectors**

Thank you